Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

TEACHING DIGITAL DESIGN WITH OPEN-SOURCE EDA

Exploring (new?) possibilities

Last Update: June 26, 2023

M.Sc. Thorsten Knoll
Prof.Dr. Steffen Reith

Computer Science
Hochschule **RheinMain**

ABOUT RESOURCES

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available
- No funding / minor support

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available
- No funding / minor support
- No students to help

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available
- No funding / minor support
- No students to help
- No suitable curriculum (computer science!)

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available
- No funding / minor support
- No students to help
- No suitable curriculum (computer science!)
- No encouragement by colleagues (“Steffen, you are a mathematician! . . .”)

ONCE UPON A TIME ...

Disclaimer: We (Thorsten and me) are computer scientists, not electrical engineers!

- Real task (Automotive industry): **Build an ignition key** based on a Zero-Knowledge protocol which is cheap and fast ($< 70ms$).
- We ended above $80ms$.

Idea: Build dedicated hardware

- No knowledge about digital design available
- No funding / minor support
- No students to help
- No suitable curriculum (computer science!)
- No encouragement by colleagues ("Steffen, you are a mathematician! ...")

Question: How can we bootstrap the process to build hardware?

WHEN ALL YOU HAVE IS ...

Secret ingredient

If you want to build a ship, don't drum up the men to gather wood, divide the work, and give orders. Instead, teach them to yearn for the vast and endless sea.

(Antoine de Saint-Exupéry)

WHEN ALL YOU HAVE IS ...

Secret ingredient

If you want to build a ship, don't drum up the men to gather wood, divide the work, and give orders. Instead, teach them to yearn for the vast and endless sea.

(Antoine de Saint-Exupéry)

Define projects / interesting stuff for students:

- Communication interfaces (UART, SPI, Ethernet)
- CPUs (**RISC-V**, Intersil RTX2010)
- Cryptography (Elliptic curves, **AES**, SHA3)
- Cellular automata (Conway's **Game of Life**)
- Fractals (**Mandelbrot sets** in real time)
- **Retro computing** (Gameboy sound generation, "Arcade games")
- Reverse engineering (Gameboy display to HDMI)

WHEN IT BREATHES, TEACH IT!

Some lessons learned. For our bootstrap, we need:

- **“Good” literature** to build things was/is hard to find (either “simulation” or “low level”).
- **Modern** and productive **synthesis tools**. Computer engineers want to automate (no IDEs but Makefiles). We need tools for **young** engineers, later they do the work (no VHDL & Verilog, no Xlinx/Altera, no Cadence)
- Cheap **takeaway hardware**, so everybody can work at home

WHEN IT BREATHES, TEACH IT!

Some lessons learned. For our bootstrap, we need:

- **"Good" literature** to build things was/is hard to find (either "simulation" or "low level").
- **Modern** and productive **synthesis tools**. Computer engineers want to automate (no IDEs but Makefiles). We need tools for **young** engineers, later they do the work (no VHDL & Verilog, no Xlinx/Altera, no Cadence)
- Cheap **takeaway hardware**, so everybody can work at home

Hence,

- use cheap FPGAs first (implicit training for mass market applications)
- students **don't want to sign NDA's** / export regulations (this is a **real killer** if digital design is mandatory in your curriculum)
- Industry asks for junior developers, but I **cannot** (do not want to) **teach secret technology**

LEARN ONE'S LESSON

25 - 30 years ago, the **situation was similar** with software:

- all flavours of **UNIX** had licence problems or were **closed source** ($\rightarrow$ adding a OS-feature / fixing bugs was difficult)
- **insufficient and expensive tooling** for developing (complex) software
- **small** developer **community**

LEARN ONE'S LESSON

25 - 30 years ago, the **situation was similar** with software:

- all flavours of **UNIX** had licence problems or were **closed source** ($\rightarrow$ adding a OS-feature / fixing bugs was difficult)
- **insufficient and expensive tooling** for developing (complex) software
- **small** developer **community**
- It was easy to implant "**Trojans**" by **untrustworthy parties**

LEARN ONE'S LESSON

25 - 30 years ago, the **situation was similar** with software:

- all flavours of **UNIX** had licence problems or were **closed source** ($\rightarrow$ adding a OS-feature / fixing bugs was difficult)
- **insufficient and expensive tooling** for developing (complex) software
- **small** developer **community**
- It was easy to implant "**Trojans**" by **untrustworthy parties**

The advent of **Linux** (and **386BSD**) created a whole **new industry**, **new ideas**, **new developers** and **new business models**.

Reason: Open-source **brings different expertise together** and **share the costs**.

LEARN ONE'S LESSON

25 - 30 years ago, the **situation was similar** with software:

- all flavours of **UNIX** had licence problems or were **closed source** (→ adding a OS-feature / fixing bugs was difficult)
- **insufficient and expensive tooling** for developing (complex) software
- **small** developer **community**
- It was easy to implant "**Trojans**" by **untrustworthy parties**

The advent of **Linux** (and **386BSD**) created a whole **new industry**, **new ideas**, **new developers** and **new business models**.

Reason: Open-source **brings different expertise together** and **share the costs**.

Clone this idea to develop **(open) hardware**!

OPEN-SOURCE-HARDWARE

For the bootstrap process, **open-source is ideal / mandadory**

- **Cheap FPGA-Takeaway** hardware for beginners
- **Open EDA-Software is available** can be modified, and the students can work on any system and any place (**No NDA**/ **license-issues**/**lawyers**)
- We can **use modern environments** (that are not necessarily better) to **play the game** according to the **rules of the students**

OPEN-SOURCE-HARDWARE

For the bootstrap process, **open-source is ideal / mandadory**

- **Cheap FPGA-Takeaway** hardware for beginners
- **Open EDA-Software is available** can be modified, and the students can work on any system and any place (**No NDA**/ **license-issues**/**lawyers**)
- We can **use modern environments** (that are not necessarily better) to **play the game** according to the **rules of the students**

Benefits for **industry**:

- The construction of **modern** hardware often **evolved** from the “old” principles.
  → **Teach the basics** using older principles
- Germany is strong in automation, mechanical engineering and chemical industry.
  We need solid and simple microcontrollers!
  → **Maybe** open source **can help more** than you **might think!**

OPEN-SOURCE-HARDWARE (II)

Benefits for **research**:

- **Full disclosure** of all research artefacts is possible (“raw data”)
- **Comparison** of research results is much **easier**

Open-Source-Principles have major benefits and are nearly mandatory for teaching; hence, they are a valuable contribution to the field **even** when they **don't use cutting edge technology!**

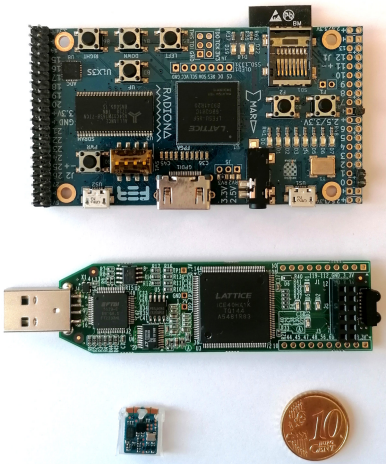
EXAMPLES

A DECADE OF MINE

FPGAs:

- 2013: Vivado 20-30GB.
  Licence files via email.
  Hardly Linux capable.
- 2015: Yosys, Lattice FPGA
  and Next-PNR.
- 2021: QuantumRISC-VM.
- Now: oss-cad-suite.
  Plenty of good and affordable boards.
  For CS students, it's more fun now!

https://quantumrisc.org/projektergebnisse_en.html
https://github.com/YosysHQ/oss-cad-suite-build/releases
https://radiona.org/ulx3s/
https://www.latticesemi.com/icestick
https://tomu.im/fomu.html

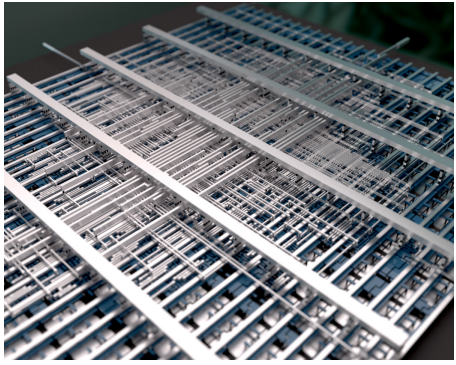


Picture by Thorsten Knoll

A DECADE OF MINE

ASICs:

- 2013: Far from reach and expensive
- 2021: Zero To Asic Course, Open-MPW, SKY130 PDK, OpenRoad and -Lane
- 2022: Multiple open-source tapeouts
- 2023: IHP130 open PDK, HEP RiscV, **Hackathon: TinyTapeOut, Hackathon: Standardcells**
- Upcoming: Teaching open-source EDA courses in CS curriculum @ HSRM.



GDS Rendering by Maximo Balestrini

https://www.zerotoasiccourse.com
https://github.com/google/skywater-pdk
https://github.com/IHP-GmbH/IHP-Open-PDK
https://github.com/The-OpenROAD-Project
https://hep-alliance.org/

SHARING SPACE AND COSTS FOR ASICS

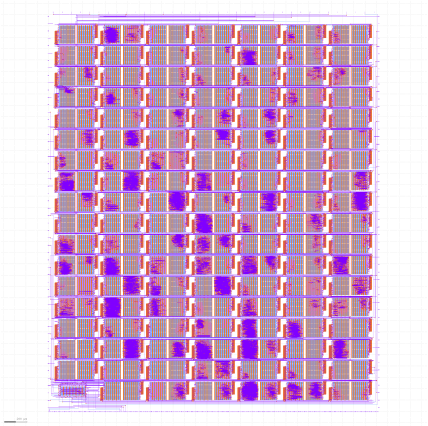
Sharing Multi Project Wafers (MPW):

- Sharing is easier with open-source.
- 1 MPW shuttle = 40 MPW projects
- 1 MPW project = 250 TinyTapeOut designs
- Get your (many) designs on a chip for small money.
- Perfect for teaching students.

It's open-source:

https://github.com/TinyTapeout
Please help: Use, modifiy, review, participate!

Matthew Venn runs TinyTapeOut here:
http://tinytapeout.com.



TinyTapeOut 03 GDS rendering by Matthew Venn

MICROCHIP DESIGN DAYS

We do **one-day digital chipdesign** Hackathons with our students.

- Jumpstart for CS undergraduates without prior knowledge in EDA.
- No Hardware description language needed (Graphical design in browser, Wokwi)
- No Tools needed (Github Actions + CI)
- Submission via TinyTapeOut
- We were the first university to work with TinyTapeOut
- Other universities are: JKU Linz, Stanford, Oklahoma state

STANDARD CELLS: VIZUALISATION

Using and contributing to open-source:

IHP130 open-source PDK layer documentation

\+ IHP130 open-source PDK GDS library

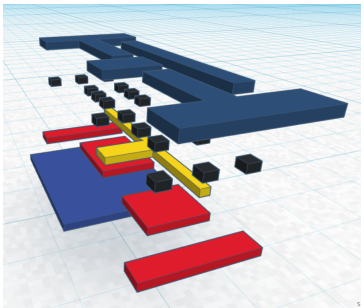
\+ gdsiistl open-source tool

———————————————-

= 3D Printable STL files

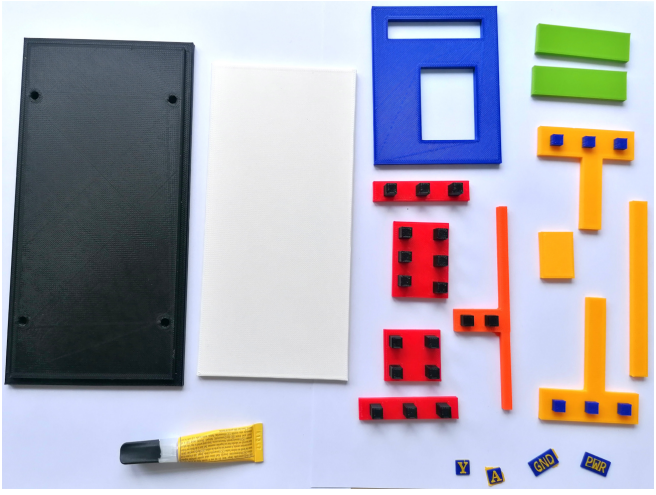
My adaption of **gdsiistl for IHP130 PDK**:
https://github.com/ThorKn/gdsiistl

Hackathon of self-created IHP130 cells as contribution to the open-source PDK: Commit ahead.



Picture by Thorsten Knoll

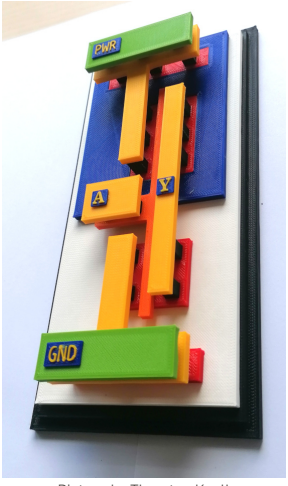
STANDARD CELLS. IHP130 INVERTER



Picture by Thorsten Knoll

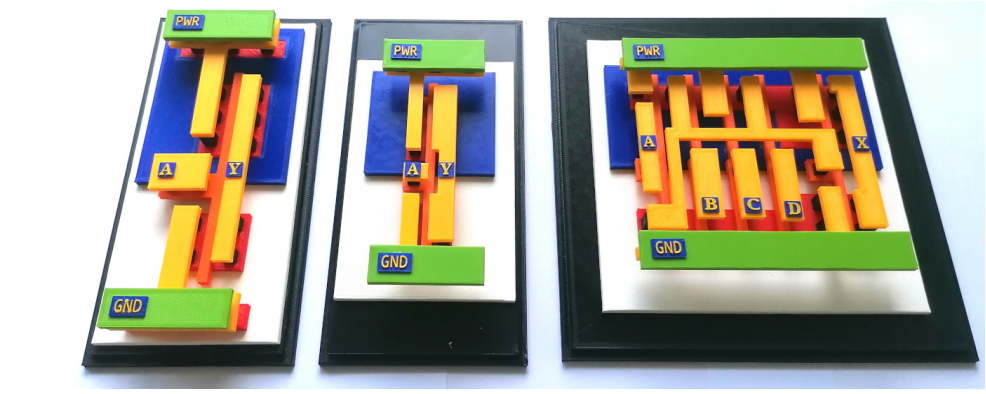
STANDARD CELLS. IHP130 INVERTER

Scale:
40000 : 1
4cm : 1um



Picture by Thorsten Knoll

STANDARD CELLS. COMPARE DIFFERENT PDKS



Picture by Thorsten Knoll

My 'How-to' blog post: **https://medium.com/@thorstenknoll/open-source-ic-cells-as-3d-prints-a-rough-how-to-guide-90a8bc8b3b57**

Q AND A

We have a few printed 3D cells with us. Talk to us, we won't carry them back home :)
Open-source is sharing!

Thank you.

Questions?

steffen.reith@hs-rm.de
thorsten.knoll@hs-rm.de